

Principal Polynomial Analysis

Valero Laparra, Sandra Jiménez,
Devis Tuia, Gustau Camps-Valls and Jesús Malo ^{*†‡}

Abstract

This paper presents a new framework for manifold learning based on a sequence of principal polynomials that capture the possibly nonlinear nature of the data. The proposed Principal Polynomial Analysis (PPA) generalizes PCA by modeling the directions of maximal variance by means of curves, instead of straight lines. Contrarily to previous approaches, PPA reduces to performing simple univariate regressions, which makes it computationally feasible and robust. Moreover, PPA shows a number of interesting analytical properties. *First*, PPA is a volume-preserving map, which in turn guarantees the existence of the inverse. *Second*, such an inverse can be obtained in closed form. Invertibility is an important advantage over other learning methods, because it permits to understand the identified features in the input domain where the data has physical meaning. Moreover, it allows to evaluate the performance of dimensionality reduction in sensible (input-domain) units. Volume preservation also allows an easy computation of information theoretic quantities, such as the reduction in multi-information after the transform. *Third*, the analytical nature of PPA leads to a clear geometrical interpretation of the manifold: it allows the computation of Frenet-Serret frames (local features) and of generalized curvatures at any point of the space. And *fourth*, the analytical Jacobian allows the computation of the metric induced by the data, thus generalizing the Mahalanobis distance. These properties are demonstrated theoretically and illustrated experimentally. The performance of PPA is evaluated in dimensionality and redundancy reduction, in both synthetic and real datasets from the UCI repository.

1 Introduction

Principal Component Analysis (PCA), also known as the Karhunen-Loève transform or the Hotelling transform, is a well-known method in machine learning, signal processing and statistics [24]. PCA essentially builds an orthogonal transform to convert a set of observations of possibly correlated variables into a set of linearly uncorrelated variables. PCA has been used for manifold description and dimensionality reduction in a wide range of applications because of its simplicity, energy compaction, intuitive interpretation, and invertibility. Nevertheless, PCA is hampered by data exhibiting nonlinear relations. In this paper, we present a nonlinear generalization of PCA that, unlike other alternatives, keeps all the above mentioned appealing properties of PCA.

^{*}Electronic version of an article published as Int. J. Neural Syst. 24(7) (2014) [DOI: 10.1142/S0129065714400073] [copyright World Scientific Publishing Company].

[†]Image Processing Laboratory (IPL), Universitat de València, Catedrático A. Escardino - 46980 Paterna, València (Spain). E-mail: {valero.laparra, jesus.malo, gustau.camps}@uv.es

[‡]This work was partially supported by the Spanish Ministry of Economy and Competitiveness (MINECO) under project TIN2012-38102-C03-01, and under a EUMETSAT contract.

1.1 Desirable properties in manifold learning

In recent years, several dimensionality reduction methods have been proposed to deal with manifolds that can not be linearly described (see [32] for a comprehensive review): the approaches proposed range from local methods [4, 45, 49, 50, 52], to kernel-based and spectral decompositions [44, 46, 53], neural networks [15, 20, 26], and projection pursuit methods [22, 27]. However, despite the advantages of nonlinear methods, classical PCA still remains the most widely used dimensionality reduction technique in real applications. This is because PCA: 1) is easy to apply, 2) involves solving a convex problem, for which efficient solvers exist, 3) identifies features which are easily interpretable in terms of original variables, and 4) has a straightforward inverse and out-of-sample extension.

The above properties, which are the base of the success of PCA, are not always present in the new nonlinear dimensionality reduction methods due either to complex formulations, to the introduction of a number of non-intuitive free parameters to be tuned, to their high computational cost, to their non-invertibility or, in some cases, to strong assumptions about the manifold. More plausibly, the limited adoption of nonlinear methods in daily practice has to do with the lack of feature and model interpretability. In this regard, the usefulness of data description methods is tied to the following properties:

1. *Invertibility of the transform.* It allows both characterizing the transformed domain and evaluating the quality of the transform. On the one hand, inverting the data back to the input domain is important to understand the features in physically meaningful units, while analyzing the results in the transformed domain is typically more complicated (if not impossible). On the other hand, invertible transforms like PCA allow the assessment of the dimensionality reduction errors as simple reconstruction distortion.
2. *Geometrical interpretation of the manifold.* Understanding the system that generated the data is the ultimate goal of manifold learning. Inverting the transform is just one step towards knowledge extraction. Geometrical interpretation and analytical characterization of the manifolds give us further insight into the problem. Ideally, one would like to compute geometric properties from the learned model, such as the curvature and torsion of the manifold, or the metric induced by the data. This geometrical characterization allows to understand the latent parameters governing the system.

It is worth noting that both properties are scarcely achieved in the manifold learning literature. For instance, *spectral* methods do not generally yield intuitive mappings between the original and the intrinsic curvilinear coordinates of the low dimensional manifold. Even though a metric can be derived from particular kernel functions [6], the interpretation of the transformation is hidden behind an implicit mapping function, and solving the pre-image problem is generally not straightforward [21]. In such cases, the application of (indirect) evaluation techniques has become a relevant issue for methods leading to non-invertible transforms [51]. One could argue that direct and inverse transforms can be alternatively derived from mixtures of local models [4]. However, the effect of these local alignment operations in the metric is not trivial. In the same way, explicit geometric descriptions of the manifold, such as the computation of curvatures, is not obvious from other invertible transforms, as autoencoders or deep networks [15, 20, 26, 27].

In this paper, we introduce the Principal Polynomial Analysis (PPA), which is a *nonlinear generalization of PCA* that still shares all its important properties. PPA is computationally easy as it only relies on matrix inversion and multiplication, and it is robust since

it reduces to a series of marginal (univariate) regressions. PPA implements a volume-preserving and invertible map. Not only the features are easy to interpret in the input space but, additionally, the analytical nature of PPA allows to compute classical geometrical descriptors such as curvature, torsion and the induced metric at any point of the manifold. Applying the learned transform to new samples is also as straightforward as in PCA. Preliminary versions of PPA were presented in [31], and applied to remote sensing in [30]. However, those conference papers did not study the analytical properties of PPA (volume preservation, invertibility, and model geometry), nor compared with approaches that follow similar logic like NL-PCA.

1.2 Illustration of Principal Polynomial Analysis

The proposed PPA method can be motivated by considering the conditional mean of the data. In essence, PCA is optimal for dimensionality reduction in a mean square error (MSE) sense *if and only if* the conditional mean in each principal component is constant along the considered dimension. Hereafter, we will refer to this as the *conditional mean independence* assumption. Unfortunately, this symmetry requirement does not apply in general, as many datasets live in non-Gaussian and/or curved manifolds. See for instance the data in Fig. 1 (left): the dimensions have a nonlinear relation even after PCA rotation (center). In this situation, the mean of the second principal component given the first principal component can be easily expressed with a parabolic function (red line). For data manifolds lacking the required symmetry, nonlinear modifications of PCA should remove the residual nonlinear dependence.

Following the previous intuition, PPA aims to remove the condition mean. Left panel in Fig. 1 shows the input $2d$ data distribution, where we highlight a point of interest, \mathbf{x} . PPA is a sequential algorithm (as PCA) that transforms one dimension at each step in the sequence. The procedure in each step consists of two operations. The *first operation* looks for the best vector for data projection. Even though different possibilities will be considered later (Section 2.3), a convenient choice for this operation is the leading eigenvector of PCA. Figure 1[middle] shows the data after this projection: although the linear dependencies have been removed, there are still relations between the first and the second data dimensions. The *second operation* consists in subtracting the conditional mean to every sample. The conditional mean is estimated by fitting a curve predicting the residual using the projections estimated by the first operation.

This step, composed of the two operations above, describes the d -dimensional data along *one* curvilinear dimension through (1) a projection score onto certain leading vector, and (2) a curve depending on the projection score. PPA differs from PCA in this second operation because it bends the straight line into a curve, thus capturing part of the nonlinear relations between the leading direction and the orthogonal subspace. Since this example is two-dimensional, PPA ends after one step. However, when there are more dimensions, the two-operations are repeated for the remaining dimensions. At the first step, the $(d-1)$ -dimensional information still to be described is the departure from the curve in the subspace orthogonal to the leading vector. This data of reduced dimension is the input for the next step in the sequence. The last PPA dimension will be the $1d$ residual which, in this example, corresponds to the residuals in the second dimension.

1.3 Outline of the paper

The paper is organized as follows. Section 2 formalizes the forward PPA transform and analytically proves that PPA generalizes PCA and improves its performance in dimensionality reduction. The objective function of PPA, its restrictions, and its computational cost are then analyzed. Section 3 studies the main properties of PPA: Jacobian, volume preservation, invertibility, and metric. In Section 4 we discuss the differences between PPA and related work. In Section 5, we check the generalization of Mahalanobis distance using the PPA metric, and its ability to characterize the manifold geometry (curvature and torsion). Finally, we report results on standard databases for dimensionality and redundancy reduction. Section 6 concludes the paper. Additionally, the appendix details a step-by-step example of the forward transform.

2 Principal Polynomial Analysis

In this section, we start by reviewing the PCA formulation as a deflationary (or sequential) method that addresses one dimension at a time. This is convenient since it allows to introduce PPA as the generalization that uses polynomials instead of straight lines in the sequence.

2.1 The baseline: Principal Component Analysis

Given a d -dimensional centered random variable \mathbf{x} , the PCA transform, \mathbf{R} , maps data from the input domain, $\mathcal{X} \subseteq \mathbb{R}^{d \times 1}$, to a response domain, $\mathcal{R} \subseteq \mathbb{R}^{d \times 1}$. PCA can be actually seen as a sequential mapping (or a set of concatenated $d-1$ transforms). Each transform in the sequence explains a single dimension of the input data by computing a single component of the response:

$$\begin{pmatrix} \mathbf{x}_0 \end{pmatrix} \xrightarrow{\mathbf{R}_1} \begin{pmatrix} \alpha_1 \\ \mathbf{x}_1 \end{pmatrix} \xrightarrow{\mathbf{R}_2} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \mathbf{x}_2 \end{pmatrix} \cdots \xrightarrow{\mathbf{R}_{d-1}} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{d-1} \\ \mathbf{x}_{d-1} \end{pmatrix}, \quad (1)$$

and hence the PCA transformation can be expressed as: $\mathbf{R} = \mathbf{R}_{d-1} \circ \mathbf{R}_{d-2} \circ \cdots \circ \mathbf{R}_2 \circ \mathbf{R}_1$. Here vectors, \mathbf{x}_p , and transforms, \mathbf{R}_p , refer to the p -th step of the sequence. Each of these elementary transforms, \mathbf{R}_p , acts only on part of the dimensions of the output of the previous transform: the residual, \mathbf{x}_{p-1} . Subscript $p=0$ refers to the input data so $\mathbf{x}_0 = \mathbf{x}$. This sequential (deflationary) interpretation, which is also applicable to PPA as we will see later, is convenient to derive most of the properties of PPA in Section 3.

In PCA, each transform \mathbf{R}_p : (1) α_p , which is the projection of the data coming from the previous step, \mathbf{x}_{p-1} , onto the unit norm vector \mathbf{e}_p ; and (2) $\mathbf{x}_p^{\text{PCA}}$, which are the residual data for the next step, obtained by projecting \mathbf{x}_{p-1} in the complement space:

$$\begin{aligned} \alpha_p &= \mathbf{e}_p^\top \mathbf{x}_{p-1} \\ \mathbf{x}_p^{\text{PCA}} &= \mathbf{E}_p^\top \mathbf{x}_{p-1}, \end{aligned} \quad (2)$$

where \mathbf{E}_p^\top is a $(d-p) \times (d-p+1)$ matrix containing the remaining set of vectors. In PCA, \mathbf{e}_p is the vector that maximizes the variance of the projected data:

$$\mathbf{e}_p = \arg \max_{\mathbf{e}} \{ \mathbb{E}[(\mathbf{e}^\top \mathbf{x}_{p-1})^2] \}, \quad (3)$$

where $\mathbf{e} \in \mathbb{R}^{(d-p+1) \times 1}$ represents the set of possible unit norm vectors. \mathbf{E}_p^\top can be any matrix that spans the subspace orthonormal

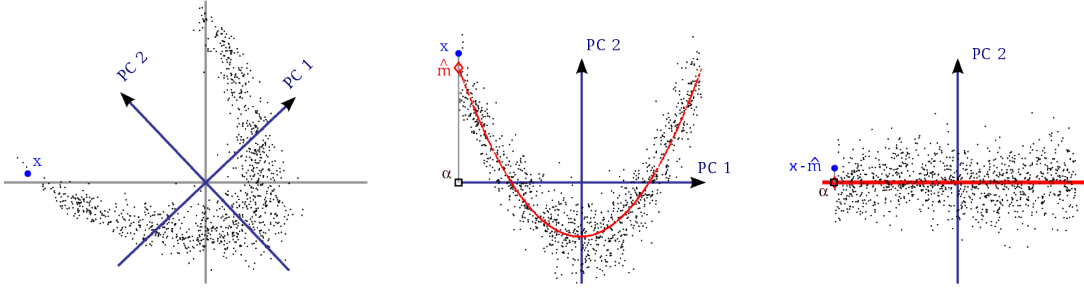


Figure 1: The *two operations* in each stage of PPA: projection and subtraction of the polynomial prediction. Left: input mean-centered data. An illustrative sample, \mathbf{x} , is highlighted. This set is not suitable for PCA because it does not fulfil the *conditional mean independence* assumption: the location of the conditional mean in the subspace orthogonal to PC1 strongly depends on PC1. Center: PCA projection (rotation) and estimation of the conditional mean by a polynomial of degree 2 (red curve) fitted to minimize the residual $\|\mathbf{x} - \hat{\mathbf{m}}\| \forall \mathbf{x}$. The black square (α) is the projection of \mathbf{x} onto PC1. The diamond (in red), $\hat{\mathbf{m}}$, in the curve represents the estimated conditional mean of \mathbf{x} predicted from the projection α . The advantage of the polynomial with regard to the straight line is that it accounts for what can be nonlinearly predicted. Right: the data after removing the estimated conditional mean (PPA solution). See the on-line paper for color figures.

to \mathbf{e}_p , and its rows contain $d - p$ orthonormal vectors. Accordingly, \mathbf{e}_p and \mathbf{E}_p fulfil:

$$\begin{aligned} \mathbf{E}_p^\top \mathbf{e}_p &= \mathbf{0} \\ \mathbf{E}_p^\top \mathbf{E}_p &= \mathbf{I}_{(d-p) \times (d-p)}, \end{aligned} \quad (4)$$

which will be referred to as the *orthonormality relations* of \mathbf{e}_p and \mathbf{E}_p in the discussion below.

In the p -th step of the sequence, the data yet to be explained is \mathbf{x}_p . Therefore, truncating the PCA expansion at dimension p implies ignoring the information contained in \mathbf{x}_p so that the dimensionality reduction error is:

$$\text{MSE}_p^{\text{PCA}} = \mathbb{E}[\|\mathbf{E}_p^\top \mathbf{x}_{p-1}\|_2^2] = \mathbb{E}[\|\mathbf{x}_p\|_2^2]. \quad (5)$$

PCA is the optimal linear solution for dimensionality reduction in MSE terms since Eq. (3) implies minimizing the dimensionality reduction error in Eq. (5) due to the orthonormal nature of the projection vectors \mathbf{e}_p and \mathbf{E}_p .

2.2 The extension: Principal Polynomial Analysis

PPA removes the conditional mean in order to reduce the reconstruction error of PCA in Eq. (5). When the data fulfill the *conditional mean independence* requirement, the conditional mean at every point in the \mathbf{e}_p direction is zero. In this case, the data vector goes through the means in the subspace spanned by \mathbf{E}_p , resulting in a small PCA truncation error. However, this is not true in general (cf. Fig. 1) and then the conditional mean $\mathbf{m}_p = \mathbb{E}[\mathbf{x}_p | \alpha_p] \neq 0$. In order to remove the conditional mean \mathbf{m}_p from \mathbf{x}_p , PPA modifies the elementary PCA transforms in Eq. (2) by subtracting an estimation of the conditional mean, $\hat{\mathbf{m}}_p$:

$$\begin{aligned} \alpha_p &= \mathbf{e}_p^\top \mathbf{x}_{p-1} \\ \mathbf{x}_p^{\text{PPA}} &= \mathbf{E}_p^\top \mathbf{x}_{p-1} - \hat{\mathbf{m}}_p \end{aligned} \quad (6)$$

Assuming for now that the leading vector, \mathbf{e}_p , is computed in the same way as in PCA, PPA only differs from PCA in the second operation of each transform \mathbf{R}_p (cf. Eq. (2)). However, this suffices to ensure the superiority of PPA over PCA. We will refer to this particular choice of \mathbf{e}_p as the *PCA-based solution* of PPA. In Section 2.3, we consider more general solutions to optimize the objective function at the cost of facing a non-convex problem. In any case,

and independently of the method used to choose \mathbf{e}_p , the truncation error in PPA is:

$$\text{MSE}_p^{\text{PPA}} = \mathbb{E}[\|\mathbf{E}_p^\top \mathbf{x}_{p-1} - \hat{\mathbf{m}}_p\|_2^2]. \quad (7)$$

Estimation of the conditional mean at step p . The conditional mean can be estimated with any regression method $\hat{\mathbf{m}}_p = g(\alpha_p)$. In this work, we propose to estimate the conditional mean at each step of the sequence using a polynomial function with coefficients $w_{p,i,j}$ and degree γ_p . Hence, the estimation problem becomes:

$$\hat{\mathbf{m}}_p = \begin{pmatrix} w_{p11} & w_{p12} & \cdots & w_{p1(\gamma_p+1)} \\ w_{p21} & w_{p22} & \cdots & w_{p2(\gamma_p+1)} \\ w_{p31} & w_{p32} & \cdots & w_{p3(\gamma_p+1)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p(d-p)1} & w_{p(d-p)2} & \cdots & w_{p(d-p)(\gamma_p+1)} \end{pmatrix} \begin{pmatrix} 1 \\ \alpha_p \\ \alpha_p^2 \\ \vdots \\ \alpha_p^{\gamma_p} \end{pmatrix}, \quad (8)$$

which, in matrix notation is $\hat{\mathbf{m}}_p = \mathbf{W}_p \mathbf{v}_p$, where $\mathbf{W}_p \in \mathbb{R}^{(d-p) \times (\gamma_p+1)}$, and $\mathbf{v}_p = [1, \alpha_p, \alpha_p^2, \dots, \alpha_p^{\gamma_p}]^\top$.

Note that when considering n input examples, we may stack them column-wise in a matrix $\mathbf{X}_0 \in \mathbb{R}^{d \times n}$. In the above mentioned *PCA-based solution*, the p -th step of the PPA sequence starts by computing PCA on \mathbf{X}_{p-1} . Then, we use the first eigenvector of the sample covariance as leading vector \mathbf{e}_p , and the remaining eigenvectors as \mathbf{E}_p . These eigenvectors are orthonormal; if a different strategy is used to find \mathbf{e}_p , then \mathbf{E}_p can be chosen to be any orthonormal complement of \mathbf{e}_p (see Section 2.3). From the projections of the n samples onto the leading vector (i.e. from the n coefficients $\alpha_{p,k}$ with $k = 1, \dots, n$), we build the Vandermonde matrix $\mathbf{V}_p \in \mathbb{R}^{(\gamma_p+1) \times n}$, by stacking the n column vectors $\mathbf{v}_{p,k}$, with $k = 1, \dots, n$.

Then, the least squares solution for the matrix \mathbf{W}_p of coefficients of the polynomial is:

$$\mathbf{W}_p = (\mathbf{E}_p^\top \mathbf{X}_{p-1}) \mathbf{V}_p^\dagger, \quad (9)$$

where \dagger stands for the pseudoinverse operation. Hence, the estimation of the conditional mean for all the samples, column-wise stacked in matrix $\hat{\mathbf{M}}_p$, is:

$$\hat{\mathbf{M}}_p = \mathbf{W}_p \mathbf{V}_p, \quad (10)$$

and the residuals for the next step are, $\mathbf{X}_p^{\text{PPA}} = \mathbf{E}_p^\top \mathbf{X}_{p-1} - \hat{\mathbf{M}}_p$.

Summarizing, the extra elements with respect to PCA are a single matrix inversion in Eq. (9) and the matrix product in Eq. (10).

Also note that the estimation of the proposed polynomial is much simpler than fitting a polynomial depending on a natural parameter such as the orthogonal projection on the curve, as one would do according to the classical Principal Curve definition [19]. Since the proposed objective function in Eq. (7) does not estimate distortions orthogonal to the curve but rather those orthogonal to the leading vector \mathbf{e}_p , the computation of the projections is straightforward and decoupled from the computation of \mathbf{W}_p . The proposed estimation in Eq. (9) consists of $d - p$ separate univariate problems only: this means that PPA needs to fit $d - p$ one-dimensional polynomials depending solely on the (easy-to-compute) projection parameter α_p . Since lots of samples $n \gg \gamma_p + 1$ are typically available, the estimation of such polynomials is usually robust. The convenience of this decoupling is illustrated in the step-by-step example presented in the appendix. Since we compute \mathbf{W}_p using least squares, we obtain three important properties:

Property 1 The PPA error does not depend on the particular selection of the basis \mathbf{E}_p if it satisfies the orthonormality relations in Eq. (4).

Proof: Using different basis \mathbf{E}'_p in the subspace orthogonal to \mathbf{e}_p is equivalent to applying an arbitrary $(d - p) \times (d - p)$ rotation matrix, \mathbf{G} , to the difference vectors expressed in this subspace in Eq. (7): $\text{MSE}_p^{\text{PPA}}(\mathbf{G}) = \mathbb{E}[(\mathbf{G}(\mathbf{E}_p^\top \mathbf{x}_{p-1} - \hat{\mathbf{m}}_p))^\top \mathbf{G}(\mathbf{E}_p^\top \mathbf{x}_{p-1} - \hat{\mathbf{m}}_p)]$. Since $\mathbf{G}^\top \mathbf{G} = \mathbf{I}$, the error is independent of this rotation, and hence independent of the basis.

Property 2 The PPA error is equal to or smaller than the PCA error.

Proof: The PPA Eqs. (7) and (9) reduce to PCA Eq. (5) in the restricted case of $\mathbf{W}_p = \mathbf{O}$. Since, in general, PPA allows for $\mathbf{W}_p \neq \mathbf{O}$, this implies that $\text{MSE}_p^{\text{PPA}} \leq \text{MSE}_p^{\text{PCA}}$. Even though the superiority of PPA over PCA in MSE terms is clearer when taking \mathbf{e}_p as in PCA, this property holds in general. If a better choice for \mathbf{e}_p is available, it would reduce the error while having no negative impact in the cost function, since it is independent from the basis \mathbf{E}_p chosen (see *Property 1* above).

Property 3 PPA reduces to PCA when using first degree polynomials (i.e. straight lines).

Proof: In this particular situation ($\gamma_p = 1, \forall p$), the first eigenvector of \mathbf{X}_{p-1} is the best direction to project onto [24]. Additionally, when using first degree polynomials, \mathbf{V}_p is very simple and \mathbf{V}_p^\dagger can be computed analytically. Plugging this particular \mathbf{V}_p^\dagger into Eq. (9), it is easy to see that $\mathbf{W}_p = \mathbf{O}$ since the data is centered and α_p is decorrelated of $\mathbf{E}_p^\top \mathbf{x}_{p-1}$. Therefore, when using straight lines \mathbf{W}_p vanishes and PPA reduces to PCA.

Finally, also note that, as in any nonlinear method, in PPA there is a trade-off between the flexibility to fit the training data and the generalization ability to cope with new data. In PPA, this can be easily controlled selecting the polynomial degree γ_p . This can be done through standard cross-validation (as in our experiments), or by using any other model selection procedure such as leave-one-out or (nested) v -fold cross-validation. Note that this parameter is also interpretable and easy to tune, since it controls the flexibility of the curves or the reduction of PPA to PCA in the $\gamma = 1$ case.

2.3 PPA cost function: alternative solutions and optimization problems

By construction PPA improves the dimensionality reduction performance of PCA when using the restricted PCA-based solution. Here we show that better solutions for the PPA cost function may exist, but unfortunately are not easy to obtain. Possible improvements would involve (1) alternative functions to estimate the conditional mean, and (2) more adequate projection vectors \mathbf{e}_p .

Better estimations of the conditional mean can be obtained with prior knowledge about the system that generated the data. For instance, if one knows that samples should follow an helical distribution, a linear combination of sinusoids could be a better choice. Even for these cases, least squares would obtain the weights of the linear combination. Nevertheless, in this work, we restrict ourselves to polynomials since they provide flexible enough solutions by using the appropriate degree. Below we show that one can fit complicated manifolds, e.g. helices, with generic polynomials. More interestingly, geometric descriptions of manifold, such as curvature or torsion, can be computed from the PPA model despite being functionally different from the actual generative model.

The selection of appropriate \mathbf{e}_p is more critical, since *Property 1* implies that MSE does not depend on \mathbf{E}_p , but only on \mathbf{e}_p . The cost function for \mathbf{e}_p measuring the dimensionality reduction error is $f(\mathbf{e})$:

$$\begin{aligned} \mathbf{e}_p &= \arg \min_{\mathbf{e}} f(\mathbf{e}) = \arg \min_{\mathbf{e}} \mathbb{E}[\|\mathbf{E}_p^\top \mathbf{x}_{p-1} - \mathbf{W}_p \mathbf{v}_p\|_2^2], \\ \text{s.t.} \quad &\mathbf{E}_p^\top \mathbf{E}_p = \mathbf{I} \\ &\mathbf{E}_p^\top \mathbf{e}_p = \mathbf{O} \\ &\mathbf{W}_p = (\mathbf{E}_p^\top \mathbf{X}_{p-1}) \mathbf{V}_p^\dagger. \end{aligned}$$

This constrained optimization does not have a closed-form solution, and one has to resort to gradient-descent alternatives. The gradient of the cost function $f(\mathbf{e})$ is:

$$\frac{\partial f}{\partial \mathbf{e}_{p_j}} = \mathbb{E} \left[\sum_{i=1}^{d-p} 2(\mathbf{E}_{p_i}^\top \mathbf{x}_{p-1} - \hat{\mathbf{m}}_{p_i}) \mathbf{W}_{p_i} \mathbf{Q} \mathbf{v}_p \mathbf{x}_{(p-1)_j} \right], \quad (11)$$

where $\mathbf{E}_{p_i}^\top$ and \mathbf{W}_{p_i} refer to the i -th rows of the corresponding matrices, $\hat{\mathbf{m}}_{p_i}$ and $\mathbf{x}_{(p-1)_j}$ are the i -th and j -th components of the corresponding vectors, and $\mathbf{Q} \in \mathbb{R}^{p \times p}$ is:

$$\mathbf{Q} = \begin{pmatrix} 0 & 1 & 0 & 0 \cdots & 0 \\ 0 & 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & p-1 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}, \quad (12)$$

In general, the PPA cost function is non-convex. The properties of $f(\mathbf{e})$ for the particular dataset at hand will determine the complexity of the problem and the accuracy of the restricted PCA-based solution. Actually, the example in Fig. 2 shows that, in general, the PCA-based solution for \mathbf{e}_p is suboptimal, and better solutions may be difficult to find given the non-convexity of the cost function. In this 2d illustration, the only free parameter is the orientation of \mathbf{e}_p . Fig. 2(b) shows the values of the error, $f(\mathbf{e})$, as a function of the orientation of \mathbf{e} . Since PCA ranks the projection by increasing variance (Eq. (3)), the PCA solution is suboptimal with respect to the one obtained by PPA with gradient descent. The first PCA eigenvector does not optimize Eqs. (7) or (11). Even worse, the risk of getting stuck into a suboptimal solution is high when using random initialization and simple gradient descent search.

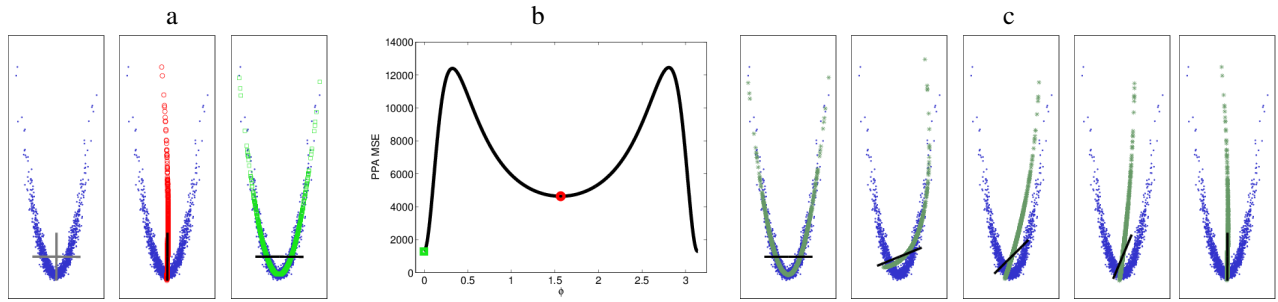


Figure 2: PPA objective is non-convex. (a) Samples drawn from a noisy parabola (blue) and the eigenvectors of the covariance matrix, PC1 and PC2 (in gray). The PPA parabolas obtained from projections onto PC1 (PCA-based solution) and onto the PC2 are plot in \circ and \square respectively. (b) Dimensionality reduction error, $f(\mathbf{e})$, for \mathbf{e}_p vectors with different orientation ϕ , where $\phi = 0$ corresponds to PC2 (\square) and $\phi = \frac{\pi}{2}$ corresponds to PC1 (\circ). (c) Fitted PPA parabolas ($*$) for a range of orientations of the corresponding \mathbf{e}_p (in black).

The results in this section suggest that the simple PCA-based solution for \mathbf{e}_p may be improved at the expense of solving a non-convex problem. According to this, in Section 4 we will present results for PPA optimized by using both the gradient descent and the PCA-based solutions. But in all cases, and thanks to *Property 2*, PPA obtains better results than PCA.

2.4 PPA computational cost

PPA is computationally more costly than PCA, which in a naïve implementation roughly scales cubically with the problem dimensionality $\mathcal{O}(d^3)$. In the case of PCA-based PPA, this cost is increased because, in each of the $d - 1$ deflationary steps, the pseudoinverse of the matrix \mathbf{V}_p has to be computed. These pseudoinverses involve $d - 1$ operations of cost $\mathcal{O}((\gamma + 1)^3)$. Therefore, in total, the cost of PCA-based PPA is $\mathcal{O}(d^3 + (d - 1)(\gamma + 1)^3)$.

If the gradient-descent optimization, Eq. (11), is used, the cost increases substantially since the same problem is solved for a number of iterations k until convergence, $\mathcal{O}(k(d^3 + (d - 1)(\gamma + 1)^3))$. The cost associated to this search may be prohibitive in many applications, but it is still lower than the cost of other generalizations of PCA: kernel-PCA scales with the number of samples, $\mathcal{O}(n^3)$, which is typically larger than the dimensionality $n \gg d$, and non-analytic Principal Curves are slow to apply since they require computing d curves per sample.

2.5 PPA Restrictions

PPA has two main restrictions that limit the class of manifolds for which PPA is well suited. First, PPA needs to fit uni-valued functions in each regression in order to ensure the transform is a bijection. This may not be a good solution when the manifold exhibits bifurcations, self-intersections, or holes. While other (non-analytical) principal curves methods can deal with such complexities [25, 42], their resulting representations could be ambiguous, since a single coordinate value would map close points, which are far in the input space. This can be in turn problematic to define an inverse function.

Secondly, PPA assumes stationarity along the principal directions as done in PCA. This is not a problem if the data follow the same kind of conditional probability density function along each principal curve. However, such condition does not hold in general. More flexible frameworks such as the Sequential Principal Curves Analysis [28] are good alternatives to circumvent this shortcoming, but at the price of a higher computational cost.

3 Jacobian, invertibility and induced metric

The most appealing characteristics of PPA (invertibility of the non-linear transform, its geometric properties and the identified features) are closely related to the Jacobian of the transform. This section presents the analytical expression of the Jacobian of PPA as well as the induced properties of volume preservation and invertibility. Then we introduce the analytical expression for the inverse and the metric induced by PPA.

3.1 PPA Jacobian

Since PPA is a composition of transforms, cf. Eq. (1), its Jacobian is the product of the Jacobians at each step:

$$\nabla \mathbf{R}(\mathbf{x}) = \prod_{p=d-1}^1 \nabla \mathbf{R}_p = \nabla \mathbf{R}_{d-1} \nabla \mathbf{R}_{d-2} \cdots \nabla \mathbf{R}_2 \nabla \mathbf{R}_1. \quad (13)$$

Therefore, the question reduces to compute the Jacobian $\nabla \mathbf{R}_p$ for each elementary transform in the sequence. Taking into account the expression for each elementary transform in Eq. (6), and the way \mathbf{m}_p is estimated in Eq. (10), simple derivatives lead to:

$$\nabla \mathbf{R}_p = \begin{pmatrix} \mathbf{I}_{(p-1) \times (p-1)} & \mathbf{0}_{(p-1) \times (d-p+1)} \\ \mathbf{0}_{(d-p+1) \times (p-1)} & \begin{pmatrix} \mathbf{e}_p^\top \\ \mathbf{E}_p^\top \end{pmatrix} - \begin{pmatrix} \mathbf{0}_{1 \times (d-p+1)} \\ \mathbf{u}_p \mathbf{e}_p^\top \end{pmatrix} \end{pmatrix}, \quad (14)$$

where $\mathbf{u}_p = \mathbf{W}_p \dot{\mathbf{v}}_p$ and $\dot{\mathbf{v}}_p = [0, 1, 2\alpha_p, \dots, \gamma_p \alpha_p^{\gamma_p-1}]^\top$. Note that the block structure of the Jacobian of each elementary transform and the identity in the top left block are justified by the fact that each \mathbf{R}_p only acts on the residual \mathbf{x}_{p-1} of the previous transform, i.e. \mathbf{R}_p does not modify the first $p - 1$ components of the previous output.

3.2 PPA is a volume-preserving mapping

Proof: The volume of any d -cube is invariant under a nonlinear mapping \mathbf{R} if $|\nabla \mathbf{R}(\mathbf{x})| = 1$, $\forall \mathbf{x} \in \mathcal{X}$ [9]. In the case of PPA, the above is true if $|\nabla \mathbf{R}_p| = 1$ for every elementary transform \mathbf{R}_p in Eq. (13). To prove this, we need to focus on the determinant of the bottom-right submatrix of $\nabla \mathbf{R}_p$, since $\begin{vmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{vmatrix} = |\mathbf{A}| |\mathbf{B}|$, where in our case \mathbf{A} is the identity matrix. Since the determinant of a matrix is the volume of the parallelogram defined by the row vectors in the matrix, the parallelogram defined by the vector \mathbf{e}_p^\top and the vectors

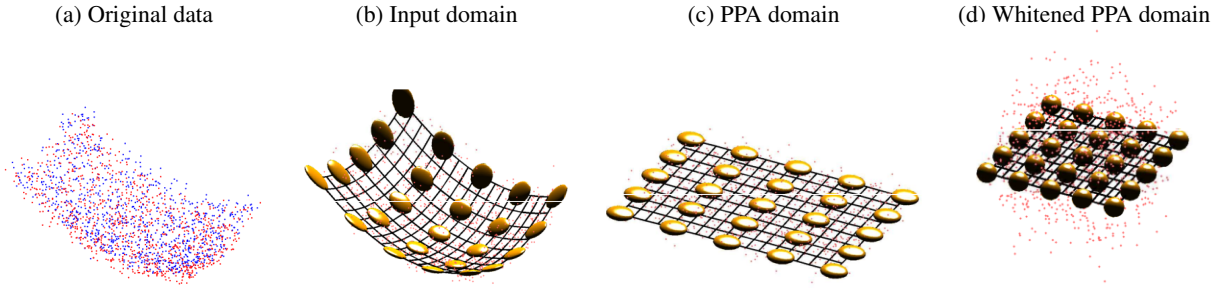


Figure 3: PPA curvilinear features and discrimination ellipsoids based on the PPA metric. (a) Non-linearly separable data. PPA results for the first class data: (b) in the input domain, (c) in the PPA domain, and (d) in the whitenened PPA domain, which is included here for the sake of comparison with the Mahalanobis metric. The curvilinear features (black grid) are computed from the polynomials found by PPA, while the unit radius spheres represent the metric induced by the whitenened PPA domain in each domain.

in \mathbf{E}_p^\top is a unit volume $(d - p + 1)$ -cube due to the orthonormal nature of these vectors. The right-hand matrix subtracts a scaled version of the leading vector, $\mathbf{u}_{pi}\mathbf{e}_p^\top$, to the vector in the i -th row of \mathbf{E}_p^\top , with $i = [1, \dots, d - p]$. Independently of weights \mathbf{u}_{pi} , this is a shear mapping of the $(d - p + 1)$ -cube defined by \mathbf{e}_p^\top and \mathbf{E}_p^\top . Therefore, after the subtraction, the determinant of this submatrix is still 1. As a result $|\nabla \mathbf{R}_p| = 1$, and hence $|\nabla \mathbf{R}(\mathbf{x})| = 1$, $\forall \mathbf{x} \in \mathcal{X}$.

Volume preservation is an appealing property when dealing with distributions in different domains. Note that probability densities under transforms depend only on the determinant of the Jacobian: $p_x(\mathbf{x}) = p_y(\mathbf{y})|\nabla \mathbf{R}(\mathbf{x})|$, for PPA $p_x(\mathbf{x}) = p_y(\mathbf{y})$. A possible use of this property will be shown in sec. 5.4 to compute the multi-information reduction achieved by the transform.

3.3 PPA is invertible

Proof: A nonlinear transform is invertible if its derivative (Jacobian) exists and it is non-singular $\forall \mathbf{x}$. This is because, in general, the inverse can be thought as the integration of a differential equation defined by the inverse of the Jacobian [14, 33]. Therefore, the volume preservation property, which ensures that the Jacobian is non-singular, also guarantees the existence of the inverse.

Here we present a straightforward way to compute the inverse by undoing each of the elementary transforms in the PPA sequence. Given that there is no loss of information in each PPA step, the inverse has perfect reconstruction, i.e. if there is no dimensionality reduction the inverted data is equal to the original one. Given a transformed point, $\mathbf{r} = [\alpha_1, \alpha_2, \dots, \alpha_{d-1}, \mathbf{x}_{d-1}]^\top$, and the parameters of the learned transform (i.e. the variables \mathbf{e}_p , \mathbf{E}_p , and \mathbf{W}_p , for $p = 1, \dots, d-1$), the inverse is obtained by recursively applying the following transform:

$$\mathbf{x}_{p-1} = \begin{pmatrix} \mathbf{e}_p & \mathbf{E}_p \end{pmatrix} \begin{pmatrix} \alpha_p \\ \mathbf{x}_p + \mathbf{W}_p \mathbf{v}_p \end{pmatrix} \quad (15)$$

3.4 PPA generalizes Mahalanobis distance

When dealing with non-linear transformations, it is useful to have a connection between the metrics (distances) in the input and transformed domains. For instance, if one applies a classification method in the transformed domain, it is critical to understand which are the classification boundaries in the original domain.

Consistently with results reported for other nonlinear mappings [13, 28, 29, 38], the PPA-induced distance in the input space follows a standard change of metric under change of coordinates [9]

and can be computed as:

$$d_{\text{PPA}}^2(\mathbf{x}, \mathbf{x} + \Delta \mathbf{x}) = \Delta \mathbf{x}^\top \mathbf{M}(\mathbf{x}) \Delta \mathbf{x}, \quad (16)$$

and the PPA-induced metric $\mathbf{M}(\mathbf{x})$ is tied to the Jacobian,

$$\mathbf{M}(\mathbf{x}) = \nabla \mathbf{R}(\mathbf{x})^\top \mathbf{\Lambda}_{\text{PPA}}^{-1} \nabla \mathbf{R}(\mathbf{x}) \quad (17)$$

and $\mathbf{\Lambda}_{\text{PPA}}$ defines the metric in the PPA domain. In principle, one can choose $\mathbf{\Lambda}_{\text{PPA}}$ depending on the prior knowledge about the problem. For instance, a classical choice in classification problems is the Mahalanobis metric [10, 37]. Mahalanobis metric is equivalent to using Euclidean metric after whitening, i.e. after dividing each PCA dimension by its standard deviation. One can generalize Mahalanobis metric using PPA by selecting a $\mathbf{\Lambda}_{\text{PPA}}$ as a matrix whose diagonal is composed by the variance of each dimension in the PPA domain. Or analogously, employing the Euclidean metric after whitening the PPA transform. Figure 3 shows an example of the unit distance loci induced by the generalized Mahalanobis PPA metric in different domains. The benefits of this metric for classification will be illustrated in Section 5.1.

4 Related Methods

The qualitative idea of generalizing principal components from straight lines to curves is not new. Related work includes approaches based on (1) non-analytical principal curves [11, 12, 28, 41, 42, 54], (2) fitting analytic curves [3, 8, 24], and (3) implicit methods based on neural networks and autoencoders [15, 20, 26] as well as reproducing kernels as in the kernel-PCA [46]. Here we review the differences between PPA and these approaches.

Non-analytic Principal Curves. In the Principal Curves literature, interpretation of the principal subspaces as d -dimensional non-linear representations is only marginally treated in [12, 41, 42]. This is due to the fact that such subspaces are not explicitly formulated as data transforms. Actually, in [42] the authors acknowledge that, even though their algorithm could be used as a representation if applied sequentially, such an interpretation was not possible at that point since the projections lacked the required accuracy. The proposed PPA is closer to the recently proposed Sequential Principal Curves Analysis (SPCA) [28] where standard and secondary principal curves [7, 19] are used as curvilinear axes to remove the non-linear dependence among the input dimensions. While flexible and interpretable, defining a transformation based on non-parametric Principal Curves (as in SPCA) has two main drawbacks: (1) it is computationally demanding since, in d -dimensional scenarios, the framework requires drawing d individual Principal Curves *per* test

sample, and (2) the lack of analytical form in the principal curves implies non-trivial parameter tuning to obtain the appropriate flexibility of the curvilinear coordinates. To resolve these issues and ensure minimal parameter tuning, we propose here to fit polynomials that estimate the conditional mean along each linear direction. We acknowledge that the higher flexibility of methods based on non-parametric Principal Curves suggests possibly better performances than PPA. However, it is difficult to prove such intuition, since, contrarily to PPA, these methods do not provide an analytic solution.

Methods fitting analytic curves. *Additive Principal Components* (APC) proposed in [8] explicitly fits a sequence of nonlinear functions as PPA. However, the philosophy of their approach differs from Principal Curves since they focus on the low variance features. In the linear case, sequential or deflationary approaches may equivalently start by looking for features that explain most or least of the variance. However, in the nonlinear APC case, the interpretation of low variance features is very different from the high variance features [8]. The high variance features identified by APC do not represent a summary of the data, as Principal Curves do. In the nonlinear case, minimizing the variance is not the same as minimizing the representation error, which is our goal. Therefore, our approach is closer to Principal Curves approaches of the previous paragraph than to APC.

Our method also presents a model and minimization of the representation error substantially different to the *Fixed Effect Curvilinear Model* in [3]. This difference in the formulation is not trivial since it makes their formulation fully d -dimensional, while we restrict ourselves to a sequential framework where $d-1$ polynomials are fitted, one at a time. Moreover, the PPA projections onto the polynomial are extracted using the subspace orthogonal to the leading vector, which makes the estimation even simpler. Additionally, their goal (minimizing the representation error in a nonlinearly transformed domain) is not equivalent to minimizing the dimensionality reduction error in the input space (as it is the case for PPA).

Neural networks and autoencoders. Neural network approaches, namely *nonlinear PCA* [15, 24, 26] and *autoencoders* [20], share many properties of PPA: they can be enforced to specifically reduce the MSE, are non-linear, invertible, and can be easily applied to new samples [48]. However, the nonlinear features are not explicit in the formulation and one is forced to use the inverse of the transformation to visualize the curvilinear coordinates of the identified low dimensional subspace. Another inconvenience is selecting the network architecture and fitting the model parameters (see [47] for a recent review), upon which the regularization ability of the network depends. The number of hidden units is typically assumed to be higher than the dimensionality of the input space, but there is still no clear way to set the network beforehand. As opposed to more explicit methods (PPA or SPCA), the curvature of the d dimensional dataset is not encoded using d nonlinear functions with different relevance, which makes the geometrical analysis difficult.

Kernel PCA. This non-linear generalization of PCA is based on embedding the data into a higher-dimensional Hilbert space. Linear features in the Hilbert space correspond to nonlinear features in the input domain [46]. Inverting the Hilbert space representation is not straightforward but a number of pre-imaging techniques have been developed [21]. However, there is a more important complication. While it is possible to obtain reduced-dimensionality representations in the Hilbert space for supervised learning [5], the KPCA formulation does not guarantee that these representations

are accurate in MSE terms in the input domain (no matter the pre-imaging technique). This is a fundamental difference with PCA (and with PPA). For this reason, using KPCA in experiments where reconstruction is necessary (as those in Section 5.3) would not be fair to KPCA.

Similarly to [16], the main motivation of PPA is finding the input data manifold that best represents that data structure in a multivariate regression problem. The above discussion suggests that the proposed nonlinear extension of PCA opens new possibilities in recent applications of linear PCA such as [1, 2, 17, 23, 40], and in cases where it is necessary to take higher order relations into account due to the nonlinear nature of the data [39].

5 Experiments

This section illustrates the properties of PPA through a set of four experiments. The first one illustrates the advantage of using the manifold-induced PPA metric for classification. The second one shows how to use the analytic nature of PPA to extract geometrical properties of the manifold. The third experiment analyzes the performance of PPA for dimensionality reduction on different standard databases. Finally, we show the benefits of the PPA volume-preserving property to compute the multi-information reduction. For the interested reader, and for the sake of reproducibility, an online implementation of the proposed PPA method can be found here:

<http://isp.uv.es/ppa.html>.

The software is written in Matlab and was tested in Windows 7 and Linux 12.4 over several workstations. It contains demos for running examples of forward and inverse PPA transforms. The code is licensed under the FreeBSD license (also known as Simplified BSD license).

5.1 Benefits of the PPA metric in classification

As presented above, the PPA manifold-induced metric provides more meaningful distance measures than the Euclidean distance or its linear Mahalanobis distance counterpart. To illustrate this, we consider k -nearest neighbors (k -NN) classification, whose success strongly depends on the appropriateness of the distance used [10].

We focus on the synthetic data in Fig. 3, where two classes are presented. They have both been generated from noisy parabolas. A cross-validation procedure on 1000 samples fitted the degree of the polynomials describing the data to $\gamma_p = 2$. Figure 4 shows the positive effect of considering PPA metric when Λ_{PPA} is a diagonal matrix with the variances of the response coefficients (i.e. generalization of the Mahalanobis distance) for k -NN classification [10]. Better performance is obtained when considering the PPA metric compared to the Euclidean or the linear Mahalanobis counterparts, especially for few training samples (Fig. 4). Moreover, the accuracy of the classifier built with the PPA metric is fairly insensitive to the number of neighbors k in k -NN, no matter the number of samples. The gain observed with the PPA metric increases with the curvature of the data distribution (bottom row of Fig. 4). Note that, with higher curvatures, the Euclidean and the linear Mahalanobis metric perform similarly poor. When a larger number of samples is available the results become roughly independent of the curvature, but even in that situation the PPA metric outperforms the others.

The generalization of the Mahalanobis metric using PPA may also be useful in extending hierarchical SOM models using more general distortion measures [35], which are useful for segmentation [34].

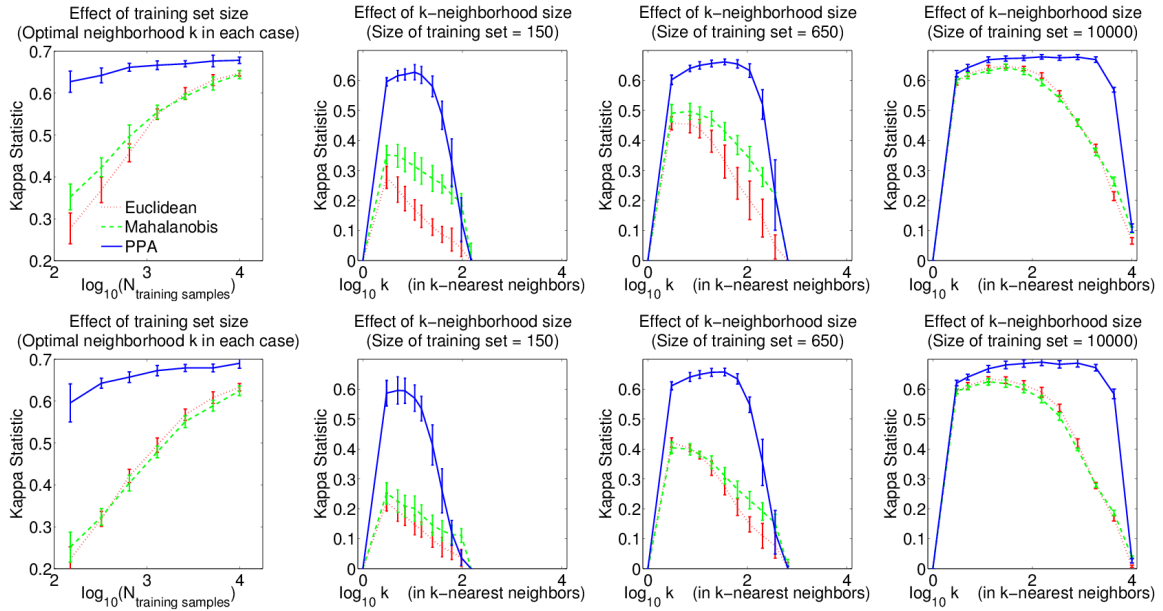


Figure 4: Effect of PPA metric in k -nearest neighbors classification for low (top) and high (bottom) curvatures.

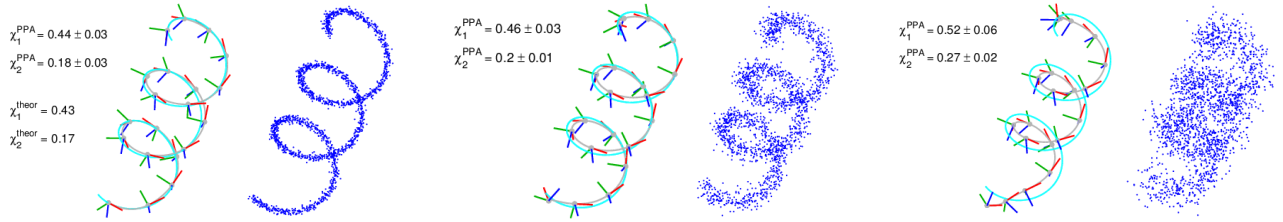


Figure 5: Geometric characterization of curvilinear PPA features in 3d helical manifolds. Scatter plots show data used to train the PPA model (1000 training and 1000 cross-validation samples) under three different noise conditions (see text). Corresponding line plots show the actual first principal curve (in cyan) and the identified first curvilinear PPA feature (in gray). The orders of the first polynomial found by cross validation were $\gamma_1 = [12, 14, 12]$, in the respective noise conditions. Lines in RGB stand for the *tangent*, *normal* and *binormal* vectors of the Serret-Frenet frame at each point of the PPA polynomial.

5.2 Differential geometry of PPA curvilinear features

According to standard differential geometry of curves in d -dimensional spaces [9], characteristic properties of a curve such as generalized curvatures χ_p , with $p = [1, \dots, d-1]$, and Frenet-Serret frames, are related to the p -th derivatives of the vector tangent to the curve. At a certain point \mathbf{x} , the vector tangent to the p -th curvilinear dimension corresponds to the p -th column of the inverse of the Jacobian.

We now use the analytical nature of PPA to obtain a complete geometric characterization of the curvilinear features identified by the algorithm. In each step of the PPA sequence, the algorithm obtains a curve (polynomial) in \mathbb{R}^d . Below we compute such characterization for data coming from helical manifolds where the comparison with ideal results is straightforward¹. Note that this is not just an illustrative exercise, because this manifold arises in real communication problems, and due to its interesting structure, it served as test case for Principal Curves Methods [42].

The first example considers a 3d helix where the Frenet-Serret frames are easy to visualize as orthonormal vectors. Figure 5 shows the first curvilinear feature identified by PPA (in gray) compared to the actual helix used to generate the 3d data (in cyan), for different

noise levels. We used $a = 2$, $b = 0.8$, and Gaussian noise of standard deviations 0.1, 0.3, and 0.6, respectively. Note that in the high noise situation, the noise scale is comparable to the scale of the helix parameters.

The tangent vectors of this first curvilinear feature (in red) are computed from the first column of the inverse of the Jacobian (using Eqs. (13) and (14)). The other components of the Frenet-Serret frames (in 3d, the *normal* and *binormal* vectors, here in green and blue), are computed from the derivatives of the tangent vector, and the generalized curvatures are given by the Frenet-Serret formulas [9]. For each of the three examples, we report the curvature values obtained by the PPA curves, as well as the theoretical values for the generating helix. Even though curvature and torsion are constant in an helix, χ_1^{PPA} and χ_2^{PPA} are slightly point-dependent. That is the reason for the standard deviation in the χ_i^{PPA} values. In this particular illustration, the effect of noise leads to a more curly helix, hence overestimating the curvatures.

In the second example, we consider a higher dimensional setting and embed 3d helices with arbitrary radius and pitch (in the [0,1] range) into the 4d space by first adding zeros in the 4th dimension, and then applying a random rotation in 4d. Since the rotation does not change the curvatures, χ_1^{theor} and χ_2^{theor} can be computed as in the 3d case, and $\chi_3^{theor} = 0$. Fig. 6 shows the alignment between χ_i^{theor} and χ_i^{PPA} for different noise levels. We also report the χ_3^{PPA} values (that should be zero). Noise implies different curvature es-

¹In 3d spaces, the two generalized curvatures that fully characterize a curve are known simply as *curvature* and *torsion*. In the case of an helix with radius, a , and pitch, $2\pi b$, the curvature and torsion are given by $\chi_1 = |a|/(a^2 + b^2)$ and $\chi_2 = b/(a^2 + b^2)$ [9].

timations along the manifold (larger variance), and, for particular combinations of a and b , noise also implies bias in the estimations: divergence from the (ideal agreement).

Also remind that the PPA formulation allows to obtain Frenet-Serret frames in more than three dimensions. However, visualization in those cases is not straightforward. For illustration purposes here we focus on the *first* PPA curvilinear dimension. Nevertheless, the same geometric descriptors (χ_i and Frenet-Serret frames) can be obtained along the curvilinear features. Estimation of curvatures from the PPA model may be interesting in applications where geometry determines resource allocation [43].

5.3 Dimensionality reduction

In this section, we first illustrate the ability of PPA to visualize high dimensional data in a similar way to Principal Volumes and Surfaces. Then, we compared the performance of PCA, PPA and nonlinear PCA (NLPCA) of [15] in terms of reconstruction error obtained after truncating a number of features.

Data. We use six databases extracted from the UCI repository². The selected databases deal with challenging real problems and were chosen according to these criteria: they are defined in the real domain, they are high-dimensional ($d \geq 9$), the ratio between the number of samples and the number of dimensions is large ($n/d \geq 40$), and they display nonlinear relations between components (which was evaluated by pre-visualizing the data). See data summary below and in table I:

- *MagicGamma*. The dataset represent traces of high energy gamma particles in a ground-based atmospheric Cherenkov gamma telescope. The available information consists of pulses left by the incoming Cherenkov photons on the photomultiplier tubes, arranged in an image plane. The input features are descriptors of the clustered image of gamma rays in an hadronic shower background.
- *Japanese Vowels*. This dataset deals with vowel identification in japanese, and contains cepstrum coefficients estimated from speech. Nine speakers uttered two Japanese vowels /æ/ successively. Linear analysis was applied to obtain a discrete-time series with 12 linear prediction cepstrum coefficients, which constitute the input features.
- *Pageblocks*. The database describes the blocks of the page layout of documents that have been detected by a segmentation process. The feature vectors come from 54 distinct documents and characterize each block with 10 numerical attributes such as height, width, area, eccentricity, etc.
- *Sat*. This dataset considers a Landsat MSS image consisting of 82×100 pixels with a spatial resolution of $80\text{m} \times 80\text{m}$, and 4 wavelength bands. Contextual information was included by stacking neighboring pixels in 3×3 windows. Therefore, 36-dimensional input samples were generated, with a high degree of redundancy.
- *Segmentation*. This dataset contains a collection of images described by 16 high-level numeric-valued attributes, such as average intensity, rows and columns of the center pixel, local density descriptors, etc. The images were hand-segmented to create a classification label for every pixel.

- *Vehicles*. The database describes vehicles through the application of an ensemble of 18 shape feature extractors to the 2D silhouettes of the vehicles. The original silhouettes come from views from many different distances and angles. This is a suitable dataset to assess manifold learning algorithms that can adapt to specific data invariances of interest.

For every dataset we normalized the values in each dimension between zero and one. We use a maximum of 20 dimensions which is the limit in the available implementation of NLPCA (<http://www.nl pca.org/>) [47]. Note that our implementation of PPA does not have this problem.

Table 1: Summary of the data-sets.

	Database	n (# samples)	d (dimension)	n/d
1	MagicGamma	19020	10	1902
2	Japanese Vowels	9961	12	830
3	Pageblocks	5473	10	547
4	Sat	6435	36	179
5	Segmentation	2310	16	144
6	Vehicles	846	18	47

PPA learning strategies. In the experiments, the alternative strategies described in Sections 2.2 and 2.3 will be referred to as: (1) **PPA**, which is the *PCA-based solution* that inherits the leading vectors e_p from PCA; and (2) **PPA GD**, which is the *gradient-descent solution* that obtains e_p via minimization of Eq. (11). In both cases, the transforms are obtained using 50% of the data, and the polynomial degree is selected automatically (in the range $\gamma \in [1, 5]$) by cross-validation using 50% of the training data.

PPA Principal Curves, Surfaces and Volumes. First we illustrate the use of PPA to visualize the "MagicGamma" data using a small number of dimensions. Figure 7 shows how the model obtained by PPA (red line and grey grids) adapts to the samples (in blue). All plots represent the same data from different points of view. Note that the relation between data dimensions cannot be explained with linear correlation.

The curve (red) in the plots corresponds to the first identified polynomial or to the data reconstructed using just one PPA dimension. The grids in the first row of Fig. 7 were computed by defining a uniform grid in the first two dimensions of the transformed PPA domain, and transforming it back into the original domain. Second row in Fig. 7 represents visualizations in three dimensions, together with grids computed inverting uniform samples in a $5 \times 5 \times 5$ cube (or 5 stacked surfaces) in the PPA domain.

The qualitative conclusion is that despite the differences in the cost function (see discussions in Sections 2.2 and 4), the first PPA polynomial (red curve) also passes *through the middle of the samples*, so it can be seen as an alternative to the Principal Curve of the data [18]. The gray grids also go *through the middle of the samples*, which suggests that not only alternative Principal Curves can be obtained with PPA, but also Principal Surfaces and Volumes [7, 18, 41]. Moreover, these surfaces and volumes help to visualize the structure of the data. This advantage can be seen clearly in the third and fourth plots of the first row, where the data manifold seems to be embedded in more than two dimensions.

Reconstruction error To evaluate the performance in dimensionality reduction, we employ the reconstruction mean square error

²The databases are available at <http://archive.ics.uci.edu/ml/datasets.html>

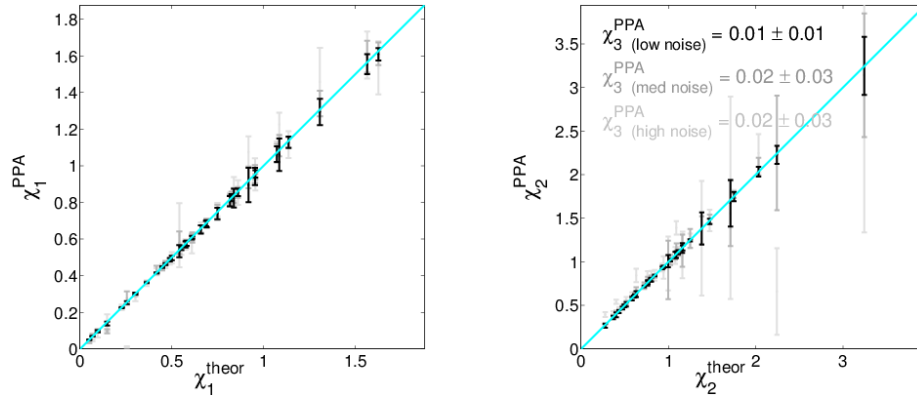


Figure 6: Geometric characterization of $4d$ helical manifolds using PPA. Prediction of generalized curvatures χ_1 (left), and χ_2, χ_3 (right) for a wide family of $4d$ helical datasets (see text for details). Darker gray stands for lower noise levels. According to the way data were generated, the theoretical value of the third generalized curvature is $\chi_3^{\text{theor}} = 0$.

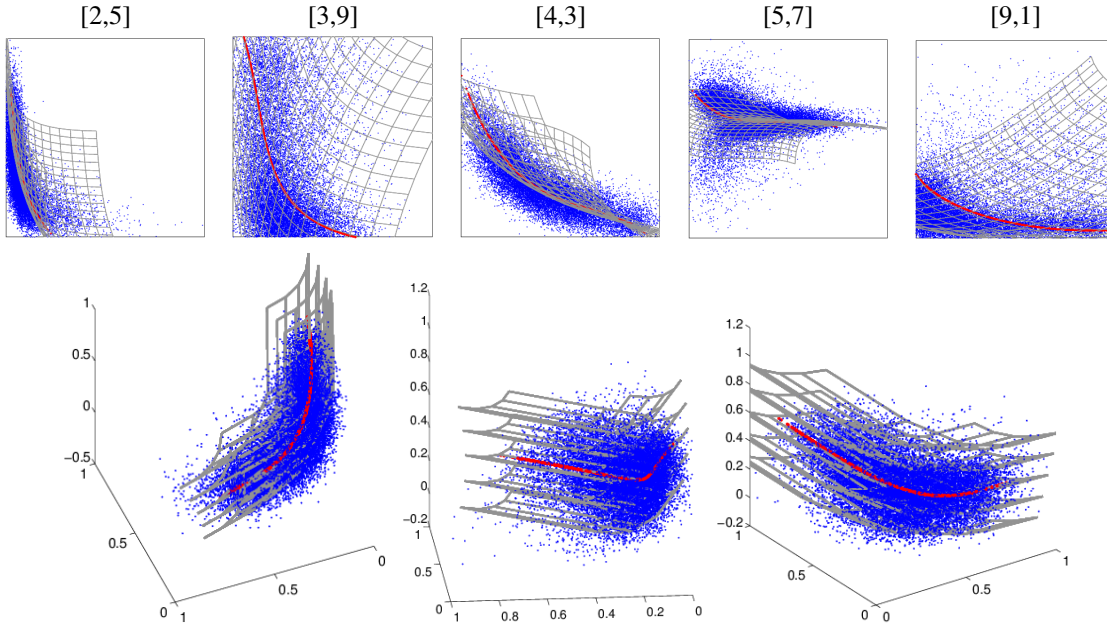


Figure 7: Principal Curves, Surfaces and Volumes using PPA. First row shows a $2d$ visualization. Titles of the panels indicate the dimensions being visualized. In each panel, are the original data (blue dots), the curve (red) is the reconstructed data (when using only one dimension) and the gray lines correspond to a grid representing the two first PPA dimensions. The second row shows $3d$ visualizations of dimensions $[3, 5, 10]$ from different camera positions. In this case, the inverted uniform grid has been constructed in the three first dimensions of the transformed domain. See text for details.

(MSE) in the original domain. For each method, the data are transformed and then inverted retaining a reduced set of dimensions. This kind of direct evaluation can be used only with invertible methods. Distortion introduced by method m is shown in terms of the relative MSE (in percentage) with regard to PCA: $\text{Rel.MSE}_m = 100 \times \text{MSE}_m / \text{MSE}_{\text{PCA}}$. Results in this section are the average over ten independent realizations of the random selection of training samples.

Figure 8 shows the results in relative MSE as a function of the number of retained dimensions. Performance on the training and test sets is reported in the top and the bottom panels respectively. Note that 100 % represents the base-line PCA error.

Several conclusions can be extracted from these results. The most important conclusion is that *PCA-based* PPA performs always better than PCA in the training set, as expected. This may not be the case with new (unobserved) test data. On the one hand, PPA is

more robust in general than NLPCA for a high number of extracted features. On the other hand, NLPCA only achieves good performance with a low number of extracted features. It is worth noting that PPA GD obtains good results for the first component in the training sets, in particular always better than PPA (as proved theoretically in sec. 2.3). Generalization ability (i.e. performance in test) depends on the method and the database. Even though a high samples per dimension ratio may help to obtain better generalization, it is not always the case (see for instance results for database “Sat”). More complex methods (as PPA GD and NLPCA) perform better in training but not necessarily in test, probably due to overfitting. More adapted schemes for training could be employed (see for instance [47]).

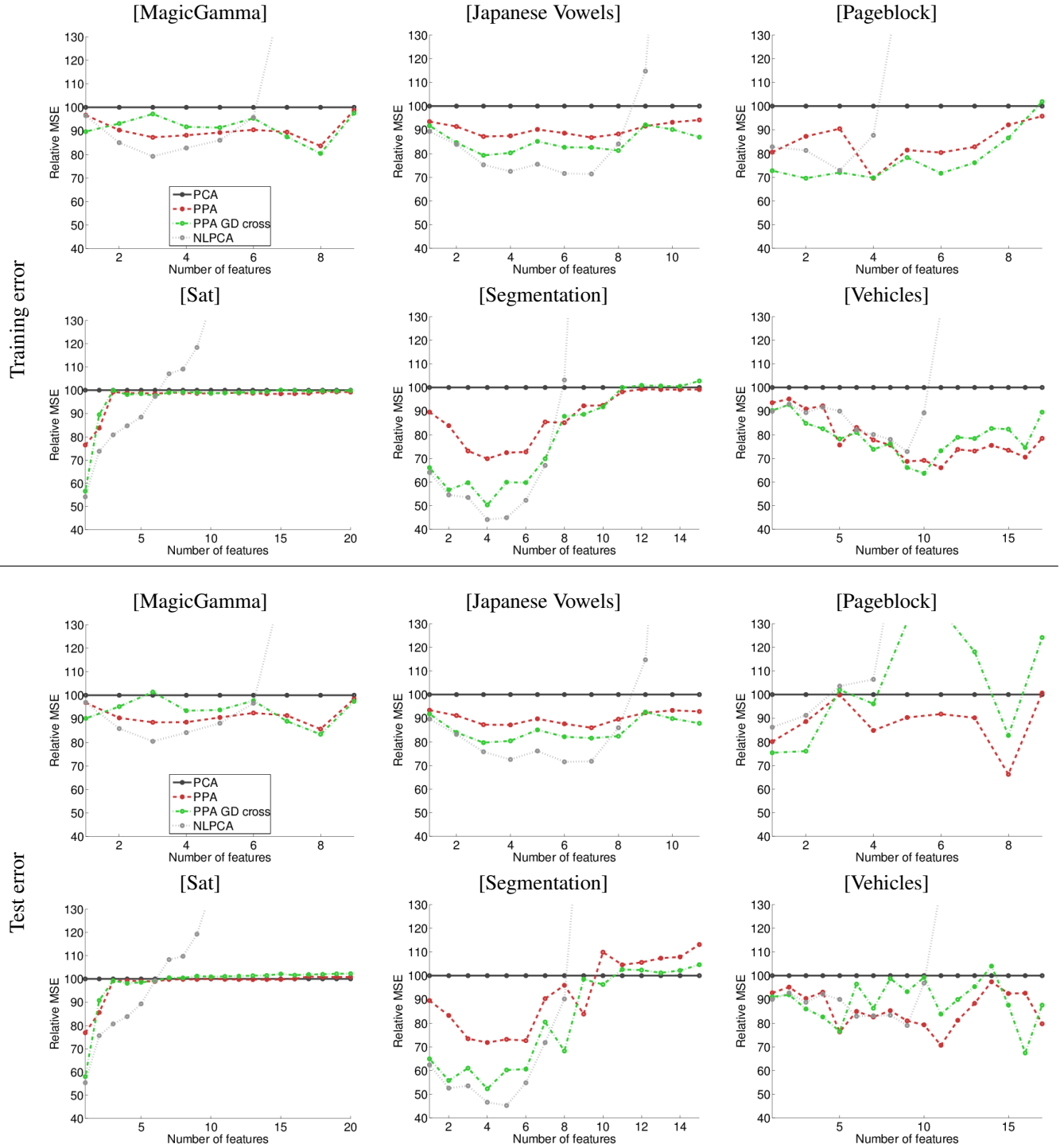


Figure 8: Relative reconstruction MSE (with regard to PCA) as a function of the retained dimensions for PCA, PPA, PPA GD and NLPCA. Top panel: results on the training data. Bottom panel: results on the test data.

Computational Cost. Table 2 illustrates the computational load for each method. The main conclusion is that PCA is the less computationally demanding, and the NLPCA the most costly, as expected. The basic PPA takes around one order of magnitude more than PCA. Although this increases the demanding time to perform an experiment it is still useful for large databases. At this point, it is worth noting that the implementation of PPA has not been optimized, it is just the straight application of the algorithm presented in Section 2. More efficient implementations could be implemented, but this is out of the scope of this work. Searching the optimal direction by gradient descent makes PPA as costly as NLPCA.

Table 2: Computational time (in min.) to learn the transform (per method and database).

	Database	Method			
		PCA	PPA	PPA GD	NLPCA
1	MagicGamma	0.0010	0.0092	142.7	80.8
2	Japanese Vowels	0.0006	0.0095	50.1	50.8
3	Pageblocks	0.0002	0.0025	7.4	20.0
4	Sat	0.0023	0.0390	68.2	122.4
5	Segmentation	0.0002	0.0065	2.5	19.8
6	Vehicles	0.0002	0.0019	0.3	9.8

5.4 Multi-information reduction

Redundancy between the features of a representation is described by the multi-information, $I(\mathbf{x})$. Therefore certain transform is suitable for efficient coding if it reduces this redundancy. Direct estimation of $I(\mathbf{x})$ is difficult since it involves Kullback-Leibler divergences between multivariate densities. However, multi-information reduction under a transform \mathbf{R} is given by [36]:

$$\begin{aligned} \Delta I &= I(\mathbf{x}) - I(\mathbf{R}(\mathbf{x})) \\ &= \sum_{j=1}^d h(\mathbf{x}^j) - \sum_{j=1}^d h(\mathbf{R}(\mathbf{x})^j) + \mathbb{E}[\log |\nabla \mathbf{R}(\mathbf{x})|], \end{aligned} \quad (18)$$

where superscript j in \mathbf{z}^j indicates its j -th feature, and $h(\mathbf{z}^j)$ is the (easy to estimate) zero-order entropy of the univariate data \mathbf{z}^j .

Therefore, multi-information reduction is particularly easy to estimate when \mathbf{R} preserves the volume because in this case $|\nabla \mathbf{R}| = 1$ so the only multivariate term in Eq. (18) vanishes. In that situation redundancy reduction just depends on comparing marginal entropies before and after the mapping, which only involves univariate densities.

Table 3 reports the multi-information reduction in bits per dimension for each database and each method. Note that NLPCA is not a volume-preserving map, and therefore its multi-information reduction can not be computed in practice. The main conclusion is that PPA obtains bigger reduction than PCA. This means that PPA obtains a representation where the dimensions of the data are more statistically independent. This is an important property of PPA when used as a preprocessing method, because one can safely apply classifiers on the projected data that assume independence between dimensions, as for instance the naive Bayes classifier.

6 Conclusions

Features extracted with linear PCA are optimal for dimensionality reduction *only* when data display a very particular symmetry. The

Table 3: Multi-information reduction (in bits per dimension) achieved by each method (bigger is better).

	Database	Method		
		PCA	PPA	PPA GD
1	MagicGamma	0.35	0.42	0.47
2	Japanese Vowels	0.38	0.45	0.49
3	Pageblock	0.16	0.23	0.25
4	Sat	1.76	1.78	1.82
5	Segmentation	1.20	1.23	1.34
6	Vehicles	1.32	1.49	1.38

proposed PPA is a nonlinear generalization of PCA that relaxes such constraints. Essentially, PPA describes the data with a sequence of curves aimed at minimizing the reconstruction error.

We analytically proved that PPA outperforms PCA in truncation error and in energy compaction. PPA also inherits all the appealing properties that make linear PCA successful: the PPA transform is computationally easy to obtain, invertible (we presented a closed-form solution for the inverse), geometrically interpretable (computable metric and curvatures), allows out-of-sample projections without resorting to approximated methods, returns a hierarchically layered representation, and does not depend on the target dimension. Additionally we showed that PPA is a volume-preserving transform, which is convenient to assess its redundancy reduction performance.

We also showed that the PPA functional is not convex. To address this problem we presented (1) a near-optimal closed-form solution based on PCA that is guaranteed to outperform PCA, and (2) the tools for a gradient descent search of the optimal solution. We analyzed the computational cost of both approaches. In the gradient descent solution the cost is very high, similar to representations based on Principal Curves, non-linear PCA, or kernel PCA. On the contrary, the cost of the PCA-based solution is only moderately bigger than PCA and clearly inferior to the above methods. Finally, results on real data showed the practical performance of PPA on dimensionality and redundancy reduction compared to PCA and non-linear PCA. In average, PPA roughly reduces a 15% both the MSE reconstruction error and the redundancy of PCA.

7 Acknowledgments

This paper has been partially supported by the Spanish MINECO under project TIN2012-38102-C03-01 and by the Swiss NSF under project PZ00P2-136827.

A Appendix: Forward PPA illustrated

Figure 9 presents a step-by-step example to illustrate how the sequence of PPA curvilinear components and projections are computed on a manifold of well-defined geometry: an helix embedded in a 3d space corrupted with additive Gaussian noise which is a usual test case in Principal Curves [42]. Data (in gray) were sampled from the same helix as in section 5.2 and noise with standard deviation 0.3. Since $d = 3$, PPA consists of a sequence of two transforms (see Eq. (1)): \mathbf{R}_1 (first row in Fig. 9) and \mathbf{R}_2 (second row). A representative sample is highlighted throughout the transform.

In this example we use the PCA-based solution. Therefore, the leading vector \mathbf{e}_1 is the first eigenvector (biggest eigenvalue) of the covariance matrix of \mathbf{x}_0 . In the example, \mathbf{e}_1 (or PC1, in orange),

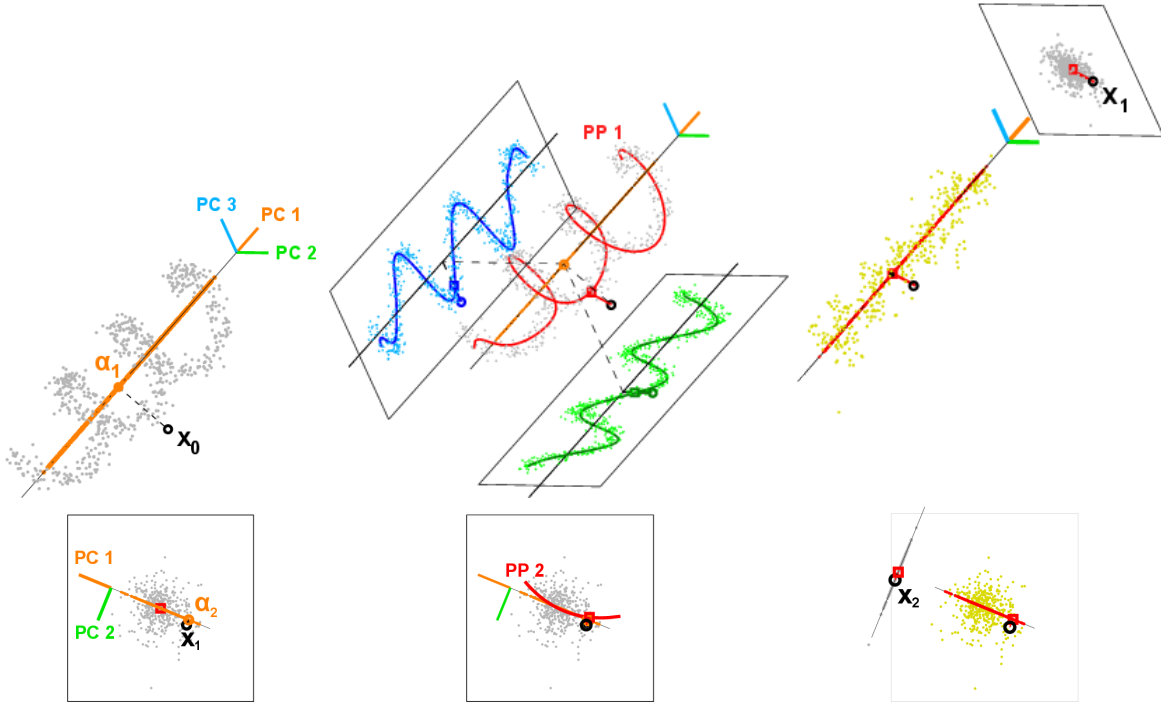


Figure 9: Forward PPA transform illustrated in a 3d example. Top row summarizes the steps in the first transform of the sequence \mathbf{R}_1 , which accounts for one curvilinear dimension and leads to a 2d residual: projection (left), polynomial fit (center), and conditional mean subtraction (right). See text for details on the symbols. Bottom row shows the equivalent steps in \mathbf{R}_2 , that leads to the final 1d residual.

and the vectors PC2 and PC3 (in green and blue respectively) constitute the basis \mathbf{E}_1 . The first PPA component, α_1 , is the projection of the data onto the first leading vector, $\alpha_1 = \mathbf{e}_1^\top \mathbf{x}_0$, in Eq. (6) (orange dots and the circle for the highlighted sample). The conditional mean, \mathbf{m}_1 , is shown decomposed in two subspaces in the top center panel. We will call \mathbf{m}_{1a} the conditional mean in the subspace spanned by \mathbf{e}_1 and PC2 (green dots), and let \mathbf{m}_{1b} be the conditional mean in the subspace spanned by \mathbf{e}_1 and PC3 (blue dots). It is obvious the strong non-linear dependence of the conditional mean with α_1 , i.e. given the value of α_1 (\mathbf{e}_1 axis -black line-) it is easy to predict the value of the data in the orthogonal subspaces (blue and green dots) using a non-linear function.

Fitting the first PPA polynomial in 3 dimensions with regard to the parameter α_1 is equivalent to fitting the polynomials in the 2d subspaces in the center plot (simple univariate regressions). The polynomials in the 2d subspaces have the coefficients $\mathbf{W}_{1a} = [w_{1a1} \ w_{1a2} \ w_{1a3} \ \dots \ w_{1a(\gamma_1+1)}]$, and equivalently, \mathbf{W}_{1b} ; which are the rows of the matrix \mathbf{W}_1 . Polynomial coefficients are easy to fit by constructing the Vandermonde matrix of degree γ_1 using α_1 , $\mathbf{v}_1 = [1 \ \alpha_1 \ \alpha_1^2 \ \dots \ \alpha_1^{\gamma_1+1}]^\top$ and applying Eq. (9). This ensures the best fitting in least squares terms. Then, we estimate \mathbf{m}_{1a} (and correspondingly \mathbf{m}_{1b}) using α_1 and the weights, Eq. (8):

$$\hat{\mathbf{m}}_{1a} = w_{1a1} + w_{1a2}\alpha_1 + w_{1a3}\alpha_1^2 \dots w_{1a\gamma_1+1}\alpha_1^{\gamma_1} \quad (19)$$

In the top center panel, the estimated conditional mean, $\hat{\mathbf{m}}_1 = [\hat{\mathbf{m}}_{1a} \ \hat{\mathbf{m}}_{1b}]^\top$, is represented by the curve (red), while the curve projected in the bottom plane (green) and the curve projected in the vertical plane (blue) represent the conditional means in the respective subspaces ($\hat{\mathbf{m}}_{1a}$ and $\hat{\mathbf{m}}_{1b}$). Once the polynomial has been fitted, we can remove $\hat{\mathbf{m}}_1$ from each sample (second line in Eq. (6)) obtaining the residuals (departures from the conditional mean) represented in the top right plot (yellow dots).

Summarizing the process in the top row, the transform \mathbf{R}_1 , the first Principal Polynomial (red curve) accounts for the first curvi-

linear dimension of the data. After \mathbf{R}_1 , we have $(d - 1) = 2$ dimensions yet to be explained: \mathbf{x}_1 , at the top right and bottom left plots. The second row of Fig. 9 reproduces the same steps in the reduced dimension residual: projection onto the first PC in the bottom left plot (orange dots), fitting the polynomial (in this case, the best cross-validation solution was a second order polynomial, represented by the curve (red) in the bottom center plot, and removing the conditional mean so that the residuals (yellow dots) are aligned, and projected in the orthogonal subspace.

References

- [1] M. Al-Naser and U. Soderstrom. Reconstruction of occluded facial images using asymmetrical principal component analysis. *Integrated Computer-Aided Engineering*, 19(3):273–283, 2012.
- [2] J. Arenas-García, K. Petersen, G. Camps-Valls, and L.K. Hansen. Kernel multivariate analysis framework for supervised subspace learning: A tutorial on linear and kernel multivariate methods. *Signal Processing Magazine, IEEE*, 30(4):16–29, 2013.
- [3] P. C. Besse and F. Ferraty. Curvilinear fixed effect model. *Comp. Stats.*, 10:339–351, 1995.
- [4] Matthew Brand. Charting a manifold. In *NIPS 15*, pages 961–968. MIT Press, 2003.
- [5] M. L. Braun, J. Buhmann, and K. Müller. On relevant dimensions in kernel feature spaces. *J. Mach. Learn. Res.*, 9:1875–1908, 2008.
- [6] C. J. C. Burges. Geometry and Invariance in Kernel Based Methods. In B. Schölkopf, C. J. C. Burges, and A. J. Smola,

- editors, *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999.
- [7] P. Delicado. Another look at principal curves and surfaces. *J. Multivar. Anal.*, 77:84–116, 2001.
- [8] D. Donnell, A. Buja, and W. Stuetzle. Analysis of additive dependencies and concavities using smallest additive principal components. *The Annals of Statistics*, 22(4):1635–1668, 1994.
- [9] B. Dubrovin, S. Novikov, and A. Fomenko. *Modern Geometry: Methods and Applications*, chapter 3: *Algebraic Tensor Theory*. Springer, NY, 1982.
- [10] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification 2nd Ed.* J. Wiley & Sons, NY, 2007.
- [11] J. Einbeck, G. Tutz, and L. Evers. Local principal curves. *Stats. & Comp.*, 15:301–313, 2005.
- [12] J. Einbeck, L. Evers, and B. Powell. Data Compression and Regression through Local Principal Curves and Surfaces. *Int. J. Neural Syst.*, 20:177–192, 2010.
- [13] I. Epifanio, J. Gutiérrez, and J. Malo. Linear transform for simultaneous diagonalization of covariance and perceptual metric matrix in image coding. *Pattern Recognition*, 36:1799–1811, 2003.
- [14] I. Epifanio and J. Malo. Differential inversion of V1 nonlinearities. Tech. Rep., Univ. Valencia, 2004.
- [15] M. Fraunholz M. Scholz and J. Selbig. *Nonlinear principal component analysis: neural networks models and applications*, chapter 2, pages 44–67. Springer, 2007.
- [16] E. García-Cuesta, I. M. Galván, A.J. de Castro. Recursive discriminant regression analysis to find homogeneous groups *Int. J. Neural Syst.*, 21(1):95–101, 2011.
- [17] S. Ghosh-Dastidar, Hojjat Adeli, and N. Dadmehr. Principal component analysis-enhanced cosine radial basis function neural network for robust epilepsy and seizure detection. *IEEE Trans. Biomedical Engineering*, 55(2):512–518, Feb 2008.
- [18] T. Hastie. *Principal curves and surfaces*. PhD thesis, Stanford University, 1984.
- [19] T. Hastie and W. Stuetzle. Principal curves. *J. Am. Stat. Assoc.*, 84(406):502–516, 1989.
- [20] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.
- [21] P. Honeine and C. Richard. The pre-image problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77–88, 2011.
- [22] P. Huber. Projection pursuit. *Annals of Statistics*, 13(2):435–475, 1985.
- [23] S. Jiménez and J. Malo. The role of spatial information in disentangling the irradiance-reflectance-transmittance ambiguity. *IEEE Trans. Geosci. Rem. Sens.*, 52(8):4881–4894, 2014.
- [24] I.T. Jolliffe. *Principal component analysis*. Springer, 2002.
- [25] B. Kegl and A. Kryzak. Piecewise linear skeletonization using principal curves. *IEEE Trans. Patt. Anal. Mach. Intell.*, 24(1):59–74, 2002.
- [26] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991.
- [27] V. Laparra, G. Camps-Valls, and J. Malo. Iterative gaussianization: from ICA to random rotations. *IEEE Trans. Neur. Net.*, 22(4):537–549, 2011.
- [28] V. Laparra, S. Jiménez, G. Camps-Valls, and J. Malo. Nonlinearities and adaptation of color vision from sequential principal curves analysis. *Neural Comp.*, 24(10):2751–88, 2012.
- [29] V. Laparra, J. Muñoz Marí, and J. Malo. Divisive normalization image quality metric revisited. *JOSA A*, 27(4):852–864, 2010.
- [30] V. Laparra, D. Tuia, S. Jiménez, G. Camps-Valls, and J. Malo. Principal polynomial analysis for remote sensing data processing. In *Geosci.Rem. Sen. Sym.*, pages 4180–4183, Jul 2011.
- [31] V. Laparra, D. Tuia, S. Jiménez, G. Camps-Valls, and J. Malo. Nonlinear data description with principal polynomial analysis. In *IEEE Workshop on Machine Learning for Signal Processing*, Spain, 2012.
- [32] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.
- [33] D.J. Logan. *Introduction to non-linear partial differential equations*. Wiley&Sons, NY, 1994.
- [34] E. López-Rubio, R.M. Luque-Baena, E. Domínguez, Foreground detection in video sequences with probabilistic self-organizing maps. *Int. J. Neural Syst.*, 21(3):225–246, 2011.
- [35] E. López-Rubio, E.J. Palomo, E. Domínguez, Bregman divergences for growing hierarchical self-organizing networks. *Int. J. Neural Syst.*, 24(4), 2014.
- [36] S Lyu and E P Simoncelli. Nonlinear extraction of ‘independent components’ of natural images using radial Gaussianization. *Neural Computation*, 21(6):1485–519, 2009.
- [37] P.C. Mahalanobis. On the generalized distance in statistics. *Proc. Nat. Inst. Sci. India*, 2(1), 1936.
- [38] J. Malo, I. Epifanio, R. Navarro, and E. Simoncelli. Nonlinear image representation for efficient perceptual coding. *IEEE Transactions on Image Processing*, 15(1):68–80, 2006.
- [39] R.J. Martis, U.R. Acharya, C.M. Lim, K.M. Mandana, A.K. Ray, C. Chakraborty Application of higher order cumulant features for cardiac health diagnosis using ECG signals. *Int. J. Neural Syst.*, 23(4), 2013.
- [40] A. Meraoumia, S. Chitroub, and A. Bouridane. 2D and 3D palmprint information, PCA and HMM for an improved person recognition performance. *Integrated Computer-Aided Engineering*, 20(3):303–319, 2013.
- [41] U. Ozertem. *Locally Defined Principal Curves and Surfaces*. PhD thesis, Dept. Sci. & Eng., Oregon Health & Sci. Univ., Sept. 2008.

- [42] U. Ozertem and D. Erdogmus. Locally defined principal curves and surfaces. *J. Mach. Learn. Res.*, 12:1249–1286, 2011.
- [43] L. Ronan, R. Pienaar, G. Williams, E.T. Bullmore, T.J. Crow, N. Roberts, P.B. Jones, J. Suckling, P.C. Fletcher. Intrinsic curvature: a marker of millimeter-scale tangential cortico-cortical connectivity? *Int. J. Neural Syst.*, 21(5): 351–366, 2011.
- [44] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.
- [45] S. T. Roweis, L. K. Saul, and G. E. Hinton. Global coordination of local linear models. In *Advances in Neural Information Processing Systems 14*, pages 889–896. MIT Press, 2002.
- [46] B. Schölkopf, A. J. Smola, and K-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comp.*, 10(5):1299–1319, 1998.
- [47] M. Scholz. Validation of nonlinear PCA. *Neural Proc. Lett.*, pages 1–10, 2012.
- [48] M. Scholz, F. Kaplan, C.L. Guy, J. Kopka, and J. Selbig. Non-linear PCA: a missing data approach. *Bioinformatics*, 21(20):3887–3895, 2005.
- [49] Y. W. Teh and S. Roweis. Automatic alignment of local representations. In *NIPS 15*, pages 841–848. MIT Press, 2003.
- [50] Joshua B. Tenenbaum, Vin Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.
- [51] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *J. Mach. Learn. Res.*, 11:451–490, 2010.
- [52] J. J. Verbeek, N. Vlassis, and B. Krose. Coordinating principal component analyzers. In *In Proc. International Conference on Artificial Neural Networks.*, 914–919. Springer, 2002.
- [53] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proc. IEEE CVPR*, 988–995, 2004.
- [54] J. Zhang, U. Krüger, X. Wang and D. Chen. A Riemannian Distance Approach for Constructing Principal Curves. In *Int. J. Neural Syst.*, 209-218, 2010.